ФУНКЦИИ УПРАВЛЕНИЯ РЕЕСТРОМ WINDOWS

**Введение**

В табл.1 сведены все функции системного реестра, а описание основных следует непосредственно после таблицы.

Таблица 1 Сводка функций системного реестра

|  |  |
| --- | --- |
| Функция | Назначение |
| RegCloseKey | Закрывает открытый ключ системного реестра |
| RegConnectRegistry | Выполняет соединение с предопределенным дескриптором системного реестра на другом компьютере |
| RegCreateKeyEx | Создает новый подключ |
| RegDeleteKey | Удаляет ключ из системного реестра |
| RegDeleteValue | Удаляет значение из системного реестра |
| RegDisablePredefinedCashe | Отключает таблицу предопределенных дескрипторов системного реестра в ключе HKCU для текущего процесса |
| RegEnumKeyEx | Перечисляет все подключи данного ключа |
| RegEnumValue | Перечисляет все значения данного ключа |
| RegFlushKey | Сразу же записывает все изменения, произведенные в системном реестре |
| RegLoadKey | Загружает раздел в корневой ключ, находящийся на вершине иерархии |
| RegNotifyChangeKeyValue | Указывает на момент изменения ключа или значения в системном реестре |
| RegOpenCurrentUser | Открывает ключ HKCU для пользователя текущей потока |
| RegOpenKeyEx | Открывает существующий ключ системного реестра с расширением Win32 |
| RegOverridePredefKey | Переопределяет переопределенный ключ системного реестра в соответствии с указанным ключом системного реестра |
| RegQueryInfoKey | Возвращает информацию о ключе |
| RegQueryMultipleValues | Выбирает тип и данные для списка имен значений |
| RegQueryValueEx | Возвращает значение (с расширенными типами данных Win32) |
| RegReplaceKey | Заменяет ключ содержимым файла при перезапуске системы |
| RegRestoreKey | Считывает содержимое раздела ранее сохраненный ключ |
| RegSaveKey | Сохраняет значения и подключи данного ключа в файле улья |
| RegSetValueEx | Присваивает ключу значение (с новыми типами данных)  |
| RegUnLoadKey | Удаляет раздел из системного реестра |

**1. Создание и открытие ключей**

Для того чтобы работать с данными реестра, приложение должно сначала создать собственный ключ или открыть ключ, созданный ранее. Для создания ключа приложению необходимо вызвать функцию RegCreateKeyEx(), которая описана в файле winreg.h так:

LONG RegCreateKeyEx (HKEY hKey, LPCSTR lpSubKey, DWORD Reserved, LPSTR lpClass, DWORD dwOptions, REGSAM samDesired, LPSECURITY\_ATTRIBUTES lpSecurityAttributes, PHKEY phkResult, LPDWORD lpdwDisposition)

Параметры:

hKey – хэндл ключа, под которым предполагается создать новый подключ.

lpSubKey – указатель на строку, завершающуюся нулевым символом в конце и содержащую имя нового ключа.

Reserved – резерв.

lpClass – указатель на строку, завершающуюся нулевым символом в конце и содержащую наименование класса нового ключа.

dwOptions – специальные опции для хранения данных, связанных с ключом. Задаются идентификатором с префиксом REG\_OPTION\_.

samDesired – доступ к ключу с требуемым уровнем защиты.

lpSecurityAttributes – атрибуты защиты ключа.

phkResult – указатель на переменную HKEY, которой присваивается хэндл нового ключа.

lpdwDisposition – указатель на переменную типа DWORD, которой присваивается значение состояния подключа (был создан или открыт).

Возвращаемое значение: если ключ создан или открыт удачно, то значение ERROR\_SUCCESS. Любое другое значение является свидетельством того, что при создании или открытии ключа встретилась ошибка.

Первый аргумент - hKey -хэндл ранее открытого ключа или одно из следующих значений:

HKEY\_CLASSES\_ROOT;

HKEY\_CURRENT\_USER;

HKEY\_LOCAL\_MACHINE;

HKEY\_USERS;

HKEY\_DYN\_DATA (Win 9x) или HKEY\_PERFORMANCE\_DATA (Win NT/2000)

При инсталляции Windows создаются пять ключей. Их имена совпадают со значениями, приведенными выше. Другими словами, эти ключи являются основой для создания иерархии ключей.

Ключи, находящиеся по иерархии ниже первого из предопределенных ключей, HKEY\_LOCAL\_MACHINE, определяют физическое состояние компьютера, включая данные о типе шины, системной памяти, инсталлированном аппаратном и программном обеспечении.

Ключи, находящиеся по иерархии ниже HKEY\_CLASSES\_ROOT, определяют типы (или классы) файлов и свойства, ассоциированные с этими классами. Свойства классов определяются только программистом. Обычно эти свойства применяются при работе приложений, использующих внедрение и связывание объектов, а также приложений, использующих среду Windows (shell applications). К примеру, при открытии файлов в Explorer'e используются свойства файлов, записанные в реестре.

Ключи, подчиненные HKEY\_USERS, определяют конфигурацию по умолчанию при подключении нового пользователя на локальной машине и конфигурацию текущего пользователя.

И наконец, ключи, подчиненные HKEY\_CURRENT\_USER, определяют установки, сделанные текущим пользователем, касающиеся переменных окружения, данных о принтерах, сетевых подключениях и т. д. Кроме этого, в этой ветви дерева хранятся установки, сделанные конкретными приложениями.

Возвращаясь к аргументам функции RegCreateKeyEx(), можно сказать, что перед созданием нового ключа необходимо продумать, в какую ветвь дерева необходимо включить новый ключ. Если новый ключ необходимо создать подчиненным ключу более низкого уровня, то определенным образом можно пройти по дереву и найти тот ключ, который необходим. Кроме этого, ключ, хэндл которого указан в первом аргументе, должен быть открыт с атрибутом доступа KEY\_CREATE\_SUB\_KEY.

Вторым аргументом - lpSubKey - является указатель на строку, содержащую имя создаваемого ключа. Создаваемый ключ будет подчиненным ключа, хэндл которого указан в первом аргументе.

Третий аргумент - Reserved -зарезервирован и должен быть равным нулю.

Четвертый аргумент - lpClass - указатель на строку, определяющую класс создаваемого ключа.

Очередной, пятый аргумент - dwOptions, определяет опции создаваемого ключа. Этот аргумент может принимать одно из значений:

* REG\_OPTION\_VOLATILE
* REG\_OPTION\_NON\_VOLATILE.

В Windows 9x первое значение не используется.

Второе значение указывает, что при перезагрузке системы значение этого ключа сохраняется, т. е. информация сохраняется в файле, а не в памяти.

Следующий, шестой аргумент - samDesired, определяет маску доступа к ключу. Этот параметр представляет собой битовую шкалу и может быть комбинацией флагов, приведенных в табл.2.

Таблица 2 Флаги, составляющие маску доступа к ключу

|  |  |
| --- | --- |
| Флаг | Описание |
| KEY\_QUERY\_VALUE  | Права запрашивать данные подключен  |
| KEY\_SET\_VALUE  | Права устанавливать данные подключен  |
| KEY\_CREATE\_SUB \_KEY  | Права создавать подключи  |
| KEY\_ENUMERATE\_SUB\_ KEY  | Права перебирать подключи  |
| KEY\_NOTIFY  | Права изменять нотификацию  |
| KEY\_CREATE LINK  | Права создавать символическую связь  |
| KEY\_READ  | (STANDARD RIGHTS READ) KEY\_QUERY\_VALUE | KEY\_ENUMERATE SUB\_KEYS | KEY\_NOTIFY |
| KEY\_WRITE  | (STANDARD RIGHTS WRITE) KEY\_SET\_VALUE | KEY\_CREATE\_SUB\_KEY |
| KEY\_EXECUTE  | KEY\_READ  |
| KEY\_ALL\_ACCESS  | (STANDARD RIGHTS ALL) KEY\_QUERY VALUE | KEY\_SET\_VALUE | KEY\_CREATE \_SUB\_KEY | KEY\_ENUMERATE\_SUB\_KEYS | KEY\_CREATE\_SUB\_KEY | KEY\_NOTIFY | KEY \_CREATE\_LINK |

Седьмой аргумент - lpSecurityAttributes - указатель на структуру типа SECURITY\_ATTRIBUTES, которая определяет атрибуты безопасности создаваемого ключа. Windows 9х не поддерживает безопасность, поэтому этот параметр игнорируется.

Туда, куда указывает восьмой аргумент - phkResult - записывается хэндл созданного ключа.

И наконец, последний, девятый аргумент - lpdwDisposition - указывает место, куда будет записана информация о том, что произошло с ключом. Дело в том, что если с помощью этой функции производится попытка создать ключ, который уже существует, то ключ не создается, а просто открывается. Поэтому приложению необходимо знать, что произошло при создании ключа.

Если ключ был создан, то в поле, определяемое lpdwDisposition, записывается значение REG\_CREATED\_NEW\_KEY.

В том случае, если ключ существовал и был открыт, записываемое значение равно REG\_OPENED\_EXISTING\_KEY.

Это поле может быть использовано и для того, чтобы узнать, не открыт ли ключ другим приложением. Открытый ключ доступен только тому приложению, которое создало его. Таким образом, если приложение открывает заведомо существующий ключ и получает в ответ значение REG\_CREATED\_NEW\_KEY, то можно сделать вывод о том, что ключ занят другим приложением.

Для ОТКРЫТИЯ подключа системного реестра с требуемым типом доступа служит функция RegOpenKeyEx(). Эта функция не создает ключ, если он не существует. Вместо этого она возвращает код ошибки.

LONG RegOpenKeyEx (HKEY hKey, LPCSTR lpSubKey, DWORD Reserved, REGSAM samDesired, PHKEY phkResult)

Параметры:

hKey – хэндл открываемого ключа.

lpSubKey – указатель на строку, завершающуюся нулевым символом в конце и содержащую имя нового ключа.

Reserved – резерв.

samDesired – доступ к ключу с требуемым уровнем защиты.

phkResult – указатель на переменную HKEY, которой присваивается хэндл нового ключа.

Возвращаемое значение: если ключ создан или открыт удачно, то значение ERROR\_SUCCESS. Любое другое значение является ошибкой.

**2. Закрытие ключей и сохранение произведенных в них изменений**

Закрывается ключ с помощью функции RegCloseKey(). При этом освобождаются любые связанные с данным ключом системные ресурсы. Закрытие ключа отнюдь не вызывает задержку операций записи, происходящих непосредственно после выполнения данной функции. Синтаксис функции следующий:

LONG RegCloseKey (HKEY hKey)

Параметры:

hKey – хэндл закрываемого ключа.

Возвращаемое значение: если ключ создан или открыт удачно, то значение ERROR\_SUCCESS. Любое другое значение является ошибкой.

Проблема. Дело в том, что данные из реестра на время работы с ними переписываются в кэш и записываются обратно на диск при выполнении функции RegFlushKey(). Чтобы данные, измененные во время работы программы, не были потеряны, то перед закрытием ключа следует их сбрасывать на диск. С другой стороны, у программиста может появиться соблазн сбрасывать данные на диск достаточно часто. Так как RegFlushKey() использует огромное количество системных ресурсов, то эту функцию нужно вызывать только в том случае, когда действительно в этом есть необходимость. Синтаксис функции следующий:

LONG RegFlushKey (HKEY hKey)

Параметры:

hKey – хэндл ключа, содержимое которого должно быть сброшено на диск.

Возвращаемое значение: если ключ создан или открыт удачно, то значение ERROR\_SUCCESS. Любое другое значение является ошибкой.

**3. Добавление данных к ключам и удаление данных из ключей**

После того, как ключ создан, возникает необходимость добавить к ключу некоторые данные, которые будут использоваться программой. Для этого нужно вызвать функцию RegSetValueEx(). Эта функция устанавливает именнованное значение любого подключа системного реестра. У данного подключа может быть множество именованных значений. Синтаксис функции следующий:

LONG RegSetValueEx (HKEY hKey, LPCSTR lpszValueName, DWORD dwReserved, DWORD dwDataType, CONST BYTE\* lpData, DWORD dwByte)

Параметры:

hKey – хэндл ключа, к которому добавляются данные.

lpszValueName – указатель на строку, завершающуюся нулевым символом в конце и содержащую имя строки добавляемых данных.

Reserved – резерв.

dwDataType – тип сохраняемых данных, который задается идентификатором (табл.4).

lpData– укзатель на данные, которые будут сохраняться в выбранном значении. Объем данных, сохраняемых в системном реестре, ограничивается используемым объемом памяти. Поэтому большие значения должны хранится в файлах, а имена этих файлов следует хранить в системном реестре.

dwByte – длина сохраняемой строки данных за исключением завершающего строку нулевого символа.

Возвращаемое значение: если ключ создан или открыт удачно, то значение ERROR\_SUCCESS. Любое другое значение является ошибкой.

Первый аргумент - хэндл ключа, к которому добавляются данные.

Второй аргумент - указатель на строку, содержащую имя добавляемых данных.

Третий аргумент зарезервирован.

Четвертый аргумент определяет тип информации, который будет сохранен в качестве данных.

Пятый аргумент является указателем непосредственно на данные, которые будут сохранены.

Шестой аргумент определяет размер данных, на которые указывает пятый аргумент. Все легко и просто, не так ли?

Различают два типа удаления:

- удаление подключа из реестра;

- удаление значений ключа реестра.

Удалить подключ из реестра можно с помощью функции RegDeleteKey(). В windows NT/2000 эта функция не будет удалять подключи и не сможет быть выполнена при наличии подключей.

В Windows 9x подключи с ее помощью удаляются. Синтаксис функции следующий:

LONG RegDeleteKey (HKEY hKey, LPCSTR lpszSubKey)

Параметры:

hKey – хэндл открытого ключа.

lpszSubKey – указатель на строку, завершающуюся нулевым символом в конце и содержащую имя удаляемого подключа.

Возвращаемое значение: если ключ создан или открыт удачно, то значение ERROR\_SUCCESS. Любое другое значение является ошибкой.

Удалить данные можно с помощью обращения к функции RegDeleteValue(). Её синтаксис следующий:

LONG RegDeleteValue (HKEY hKey, LPCSTR lpszValueName)

Параметры:

hKey – хэндл открытого ключа.

lpszValueName – указатель на строку, завершающуюся нулевым символом в конце и содержащую имя удаляемого значения.

Возвращаемое значение: если ключ создан или открыт удачно, то значение ERROR\_SUCCESS. Любое другое значение является ошибкой.

**4. Выборка данных из реестра**

Если прикладной программе нужно осуществить выборку данных из реестра, то для начала программа должна определить, из какой ветви дерева реестра ей нужно выбрать данные. Естественно, что никаких функций для этого нет. При написании программы программист должен сам позаботиться об этом.

После того как решение принято, начинается второй этап. Программа должна перебирать все ключи в этой ветви до тех пор, пока не найдет нужный ключ. Для этого приложение может воспользоваться функцией RegEnumKeyEx(). Эта функция при каждом вызове выбирает информацию об одном ключе. Для перечисления подключей в приложении следует первоначально вызвать функцию RegEnumKeyEx() с установленным значением 0 параметра dwIndex и вызывать эту функцию до тех пор, пока не будет возвращено значение ERROR\_NO\_MORE\_ITEMS. Синтаксис функции следующий:

LONG RegEnumKeyEx (HKEY hKey, DWORD dwIndex, LPSTR lpName, LPDWORD lpReserved, LPSTR lpClass, LPDWORD lpcbClass, PFILETIME pftLastChanged)

Параметры:

hKey – хэндл ключа, подключи которого подлежат перечислению.

dwIndex –индекс подключа для доступа к нему.

lpName – указатель на буфер, в который поступает имя перечисляемого подключа.

lpcbName – размер буфера lpName.

lpReserved – резерв.

lpClass – указатель на буфер, в который поступает имя класса подключа.

lpcbClass – размер буфера lpcbName

lpftLastWriteTime – указатель на переменную типа FILETIME, которой присваивается дата и время последней операции записи в данный подключ. Структура FILETIME содержит 64-разрядное значение, которое представляет собой число тактов через каждые 100 нс, начиная с 1 января 1601 г.

Возвращаемое значение: если ключ создан или открыт удачно, то значение ERROR\_SUCCESS. Любое другое значение является свидетельством того, что при создании или открытии ключа встретилась ошибка.

Если возвращается значение ERROR\_NO\_MORE\_ITEMS, то это означает, что подключей у данного ключа больше нет.

Функция перебора объектов нам встречается впервые. Давайте сначала рассмотрим аргументы этой функции, а потом поговорим о том, что происходит при переборе ключей.

Первый аргумент - это хэндл ключа, подчиненные ключи которого будут перебираться в поисках нужного ключа.

Второй аргумент - dwIndex - является индексом требуемого подключа.

Третий аргумент - lpName - указывает на буфер, в который будет записано имя ключа.

Четвертый аргумент - lpcbName - определяет размер этого буфера в байтах.

Пятый аргумент, как следует из его названия - lpReserved - зарезервирован для использования в будущем и должен быть равным NULL.

Шестой аргумент - lpClass - должен указывать на буфер, в котором после завершения работы функции будет содержать имя класса подключа. Если это имя программе не требуется, то этот аргумент должен быть равным NULL.

Размер этого буфера определяется седьмым аргументом - lpcbClass.

Восьмой аргумент -lpftLastWriteTime - после завершения работы функции содержит время последнего обновления данного подключа.

Для того чтобы перебрать подключи, приложение должно сначала вызвать функцию RegEnumKeyEx() со вторым аргументом (dwIndex), равным нулю (поиск начинается с начала дерева). Если искомый ключ найден с первой попытки, то приложению повезло. В противном случае необходимо dwIndex увеличить на единицу и снова обратиться к функции. Так необходимо делать до тех пор, пока не будет найден искомый ключ или функция не вернет значение ERROR\_NO\_MORE\_ITEMS. Естественно, что поиск можно производить и в обратном порядке. Для того чтобы поиск мог быть нормально осуществлен, ключ, хэндл которого указан первым аргументом, должен быть открыт с правом доступа KEY\_ENUMERATE\_SUB\_KEYS.

Получить полную информацию о подключе можно с помощью функции RegQueryInfoKey(), аргументы которой в лекции рассматривать не будем. Допустим нужный нам подключ найден. Тогда в этом подключе необходимо найти нужные данные. Способ поиска точно такой же, как и в предыдущей функции. Для поиска необходимо перебрать все данные, связанные с подключом. Чтобы произвести этот перебор, обычно используется функция RegEnumValue(), описание которой, приведенное ниже, можно найти в файле winreg.h:

LONG RegEnumValue (HKEY hKey, DWORD dwIndex, LPSTR lpValueName, LPDWORD lpcbValueName, LPDWORD lpReserved, LPDWORD lpType, LPBYTE lpData, LPDWORD lpcbData)

Параметры:

hKey – хэндл ключа, значения которого подлежат перечислению.

dwIndex –индекс перечисляемого значения.

lpValueName – указатель на буфер, в который поступает имя перечисляемого значения.

lpcbValueName – размер буфера lpValueName.

lpReserved – резерв.

lpType – указатель на значение типа DWORD, которое принимает тип данных, присущи перечисляемому значению.

lpData – указатель на буфер, в который поступает содержимое перечиляемого значения.

lpcbData – указатель на переменную, которая содержит число байт в буфере lpData.

Возвращаемое значение: если ключ создан или открыт удачно, то значение ERROR\_SUCCESS. Любое другое значение является свидетельством того, что при создании или открытии ключа встретилась ошибка.

Если возвращается значение ERROR\_NO\_MORE\_VALUES, то это означает, что подключей у данного ключа больше нет.

Порядок использования этой функции полностью совпадает с порядком использования функции RegEnumKeyEx(), поэтому стану на нем останавливаться. Опишу только аргументы этой функции.

Понятно, что hKey - это хэндл ключа, которому принадлежит подключ, индекс которого представлен вторым аргументом - dwlndex.

Следующий аргумент – указатель на буфер, в который будет записано имя подключа. lpcbValueName определяет размер этого буфера. Аргумент lpReserved зарезервирован и должен быть равным NULL. Последние три аргумента определяют класс подключа, указатель на буфер, в который будут записаны эти данные и размер буфера. После возврата функции предпоследний аргумент содержит число записанных данных.

Таким образом, работа с системным реестром представляет собой достаточно удобный механизм доступа к установленным системным ресурсам, программам и оборудованию. Существующие стандартные значения параметров реестра документированы в справочной информации, исследование которой будет производится на практических занятиях.